BAB II
DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian sebelumnya yang sejenis telah dilakukan pada tahun 2019 dengan judul "Rancangan Sistem Informasi Pengajuan dan Pelaporan Tunjangan Kinerja Kementerian Keuangan Menggunakan Metode *Prototype*". Sistem ini bertujuan untuk membantu Biro Perencanaan dan Keuangan Kementerian Keuangan dalam mengelola tunjangan kinerja. Proses pengajuan dan pelaporan pembayaran tunjangan kinerja selama ini masih menggunakan cara manual sehingga membutuhkan waktu yang cukup lama dari satker sampai ke kementerian. Metode yang digunakan dalam pengembangan sistem adalah *prototyping*. Selain mempermudah proses pengajuan dan pelaporan tunjangan kinerja, aplikasi juga dirancang agar kementerian dapat memonitor progres pengajuan dan pelaporan pada berbagai tingkatan baik unit eselon I, kanwil maupun satker[4].

Penelitian sejenis yang berikutnya juga telah dilakukan pada tahun 2020 dengan judul "Rancang Bangun Sistem Informasi Pengajuan Anggaran dan Laporan Kegiatan Berbasis *Android* (Studi Kasus: Universitas Trilogi)". Sistem ini bertujuan untuk menerapkan sistem Aplikasi Pengajuan Anggaran Kegiatan dan Laporan Kegiatan yang terdigitalisasi dengan baik. Sistem ini dibangun menggunakan metode *waterfall, mysql,* dan menggunakan metode perancangan sistem *Entity Relationship Diagram* (ERD) dan *Data Flow Diagram* (DFD). Aplikasi Pengajuan Anggaran dan Laporan Kegiatan berbasis *Android* ini di harapkan pengguna jadi lebih mudah dalam proses pengajuan anggaran dan laporan kegiatan. Aplikasi ini juga terdapat beberapa user level, yaitu Unit Kerja, Atasan Langsung (Dekan), Wakil Rektor 1 dan 2, Rektor dan Bagian Keuangan. Masing-masing aktor mempunyai hak akses yang berbeda dalam melakukan aktivitas di aplikasi tersebut[5].

Selanjutnya penelitian lainnya telah dilakukan pada tahun 2020 dengan judul "Sistem Informasi Pengajuan Cuti dan Izin Berbasis *Web*". Sistem ini bertujuan untuk membantu aktifitas pengaturan permohonan cuti dan izin staf karyawan dan dosen. Metode yang di gunakan dalam pengembangan sistem ini adalah metode *Rapid*

Application Development (RAD). Terdapat tiga fase dalam RAD yaitu *planning, design workshop* dan *implementation* [6].

Berdasarkan Penelitian di atas dapat disimpulkan sistem yang akan saya kembangkan berfokus pada sistem informasi pengajuan untuk dapat membantu pemohon dalam mengajukan bantuan. Sistem ini berbasis *website* dan akan terintegrasi dengan notifikasi pesan instan.

2.2 Landasan Teori

Landasan teori berisi hal-hal atau teori-teori yang berkaitan dengan permasalahan dan ruang lingkup permasalahan sebagai landasan dalam pembuatan laporan ini.

2.2.1 Sistem Informasi

Sistem merupakan bagian-bagian yang saling berkaitan yang beroprasi Bersama untuk mencapai beberapa sasaran atau maksud.

Informasi ialah data yang ditangani menjadi sebuah desain yang lebih bermanfaat dan lebih kritis bagipenerima manfaat. Sumber informasi akan menjadi data. Informasi realitas yang menggambarkan peristiwa nyata. Peristiwa adalah sesuatu yang terjadi pada waktu tertentu. Informasi akan menjadi suatu desain yang signifikan untuk manfaat dan memilik inilai nyata yang dapat dirasakan dalam keputusan saat ini atau masa depan.

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang melakukan pengelolaan transaksi harian, mendukung operasi bersifat manajerial, dan kegiatan stratefi dari suatu organisasi yang menyertakan pihak luar tertentu dalam bentuk laporan[7]. Model umum sebuah sistem secara sederhana adalah input, proses, dan output. Suatu sistem juga memiliki karakteristik atau sifat-sifat tertentu yang mencirikan bahwa hal tersebut dikatakan sebagai sebuah sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut[8]:

a) Komponen Sistem

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling bekerja untuk sama membentuk satu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu subsistem. Setiap subsistem memiliki sifat dari sistem yang menjalankan suatu fungsi tertentu yang mempengaruhi proses sistem secara keseluruhan. Suatu sistem juga dapat mempunyai sistem yang lebih besar atau sering disebut “supra sistem”.

b) Batas Sistem

Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lain atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisahkan.Lingkungan Luar Sistem

Lingkungan luar dari suatu sistem adalah apapun di luar batas dari sistem yang mempengaruhi operasi sistem.

c) Penghubung Sistem

Media yang menghubungkan sistem dengan subsistem lainnya disebut penghubung sistem atau antarmuka sistem. Penghubung sistem ini memungkinkan sumber daya mengalir dari satu subsistem ke subsistem yang lainnya. Bentuk output dari satu subsistem akan menjadi input ke subsistem lain melalui penghubung sistem tersebut. Dengan demikian, dapat terjadi integrasi sistem yang membentuk satu kesatuan.

d) Masukkan Sistem

Energi yang dimasukkan ke dalam sistem disebut masukan sistem atau input sistem, yang dapat berupa pemeliharaan (maintenance input) dan sinyal (signal input). Misalnya, dalam unit sistem komputer, "program" adalah maintenance input yang digunakan untuk mengoperasikan komputer dan "data" adalah signal input untuk diproses menjadi informasi.

b) Keluaran Sistem

Keluaran sistem dapat berupa energi diproses dan diklasifikasikan menjadi keluaran yang berguna. Keluaran Ini merupakan masukan untuk subsistem lain seperti sistem informasi, maka hasil keluarannya adalah informasi. Informasi ini dapat digunakan sebagai input untuk pengambilan keputusan atau hal lain yang menjadi masukan bagi subsistem lainnya.

c) Pengolah Sistem

Suatu sistem dapat memiliki proses yang akan mengubah input menjadi output, contohnya adalah sistem informasi audit internal. Sistem ini akan memproses data audit internal menjadi laporan yang dibutuhkan oleh pengawas atau auditor.

d) Sasaran Sistem

Suatu sistem memiliki tujuan dan sasaran yang pasti dan deterministik. Jika suatu sistem tidak memiliki target maka operasi sistem tidak berguna. Sebuah sistem dikatakan berhasil jika mencapai tujuan atau target yang direncanakan.

Kualitas Informasi merupakan kualitas keluaran sistem yang berbentuk informasi yang dihasilkan sistem informasi yang dipakai. Kualitas Informasi juga bisa dibilang sebagai pengukuran kualitas sebuah sistem informasi, jika sistem informasi yang menghasilkan informasi yang akurat, relevan, dan tepat waktu maka akan berdampak positif terhadap kepuasan user.

2.2.2 Web

Web adalah sebuah sistem dengan informasi yang disajikan dalam bentuk teks, gambar, suara, dan lain-lain yang tersimpan dalam dalam sebuah server Web Internet yang disajikan dalam bentuk hiperteks. Informasi Web dalam bentuk teks umumnya ditulis dalam format HTML *(Hypertext Markup Language)*. Informasi lainnya disajikan dalam bentuk grafis (dalam format GIF,JPG,PNG), suara (dalam forma AU,WAV), dan objek multimedia lainnya (seperti MIDI, Shockwave, Quicktime Movie, 3D World) [9].

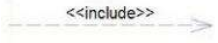
2.2.3 Laravel

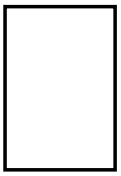
Laravel dirilis dibawah lisensi MIT dengan kode sumber yang sudah disediakan oleh Github, sama seperti framework yang lain, Laravel dibangun dengan konsep MVC *(Model-Controller-View),* kemudian Laravel dilengkapi juga *command line tool* yang bernama "Artisan" yang bisa digunakan untuk *packaging bundle* dan *instalasi bundle* melalui *command prompt.* Maka dari itu banyak programmer PHP menggunakan *Framework* Laravel karena menekankan kesederhanaan dan fleksibilitas pada desainnya[10].

2.2.4 Use Case Diagram

Use case atau *diagram use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut pada Tabel 2.4 adalah simbol-simbol yang ada pada diagram *use case*[11]:

Tabel 2. 1 *Simbol-simbol* Use Case Diagram

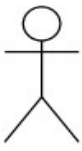
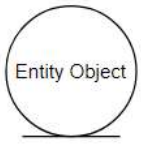
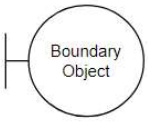
No	Simbol	Nama	Keterangan
1		*Use case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suat hasil yang terukur bagi suatu aktor.
2		*Actor*/aktor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
3		*Generalizatio n*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk *(ancestor)*.
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang dberikan.
6		*Association*	Menghubungkan antara objek satu dengan objek lainya.

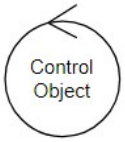
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.

2.2.5 Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambar *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat *sequence diagram* juga dibutuhkan untuk melihat skenario yang pada *use case*. Berikut pada Tabel 2.5 adalah simbol-simbol yang ada pada *sequence diagram*[11]:

Tabel 2. 2 *Simbol-simbol* Sequence Diagram

No	Simbol	Nama	Keterangan
1		*Actor*	Menggambarkan *user* atau pengguna.
2		*Entity Class*	Menggambarkan hubungan yang akan dilakukan.
3		*Boundary Class*	Menggambarkan sebuah form

4		*Control Class*	Menghubungkan antara *boundary* dengan tabel.
5		*Lifeline*	Menggambarkan hubungan kegiatan yang akan dilakukan.
6		*Massage*	Menggambarkan pengiriman pesan.

2.2.6 Basis Data

Basis data adalah kumpulan *file -file* yang mempunyai kaitan antara satu *file* dengan *file* lain sehingga membentuk satu bangun data untuk menginformasikan suatu perusahaan instansi, dalam bahasan tertentu.

MySQL adalah sebuah program *database server* yang mampu menerima dan mengirimkan datanya dengan sangat cepat, multi *user* serta menggunakan perintah standar *Structured Query Language* (SQL). MySQL memiliki dua bentuk lisensi, yaitu *Free Software* dan *Shareware*. MySQL yang biasa kita gunakan adalah MySQL *Free SoftwareI* yang berada dibawah Lisensi *General Public License* (GPL).

MySQL juga dapat didefinisikan sebagai sebuah *database server*, dapat juga berpersan sebagai *client* sehingga sering disebut *database client / server* yang *open source* dengan kemampuan dapat berjalan baik di Operasi Sistem maupun dengan *Platform Windows* maupun *Linux*[3].

MySQL dikembangkan oleh sebuah perusahaan Swedia bernama MySQL AB, yang pada saat itu bernama TcX Data Konsult AB sekitar tahun 1994-1995. MySQL sudah ada sejak 1979. MySQL termasuk jenis *Relation Database Management System* (RDBMS) digunakan oleh banyak portal-portal internet sebagai basis data dari informasi yang ditampilkan pada situs *web*. Kepopuleran MySQL dimungkinkan karena kemudahannya untuk digunakan, cepat secara kinerja *query*, dan mencukupi untuk kebutuhan basis data perusahaan-perusahaan skala

menengah dan kecil. Istilah seperti tabel, baris, dan kolom tetap digunakan dalam MySQL. Sebuah basis data yang terdapat pada MySQL mengandung satu atau beberapa tabel yang terdiri dari sejumlah baris dan kolom[3].

MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah suatu bahasa *(language)* yang digunakan untuk mengakses data di dalam sebuah *database* relasional. *SQL* sering juga disebut dengan istilah query, dan bahasa *SQL* secara praktiknya digunakan sebagai bahasa standar untuk manajemen *database* relasional. Hingga saat ini hampir seluruh *server database* atau software *database* mengenal dan mengerti bahasa *SQL*.Dalam penggunaan *SQL* terdapat beberapa perintah yang berguna untuk mengakses dan memanajemen data yang terdapat dalam *database*. Secara garis besar, *SQL Server* mempunyai 3 (Tiga) jenis perintah *SQL* yaitu:

2.2.7 Data Definition Language (DDL)

DDL adalah sub perintah dari bahasa SQL yang digunakan untuk membangun kerangka sebuah *database*, dalam hal ini *database* dan tabel. Terdapat tiga perintah penting dalam DDL, yaitu:

a. *CREATE*
   Perintah ini digunakan untuk membuat, termasuk di dalamnya membuat *database* baru, tabel baru view baru, dan kolom baru. Contoh: *CREATE DATABASE* nama_*database*.
b. *ALTER*
   Perintah *ALTER* berfungsi untuk mengubah struktur tabel yang telah dibuat. Mencakup di dalamnya mengubah nama tabel, menambah kolom, mengubah kolom, menghapus kolom, dan memberikan atribut pada kolom. Contoh: *ALTER TABLE* nama_tabel *ADD* nama_kolom *datatype*.
c. *DROP*
   Perintah *DROP* berfungsi untuk menghapus *database* atau tabel. Contoh: *DROP DATABASE* nama *database*.

2.2.8 Data Manipulation Language (DML)

DML adalah sub perintah dari bahasa SQL yang digunakan untuk memanipulasi data dalam *database* yang telah dibuat. Terdapat 4 (Empat) perintah penting dalam DML, yaitu:

a. *INSERT*

Perintah ini digunakan untuk memasukkan data baru ke dalam sebuah tabel. Perintah ini tentu saja bisa dijalankan ketika *database* dan tabel sudah dibuat. Contoh: *INSERT* INTO nama_tabel *VALUES* (data1, data2, dst…);

b. *SELECT*
Perintah ini digunakan untuk mengambil dan menampilkan data dari tabel atau bahkan dari beberapa tabel dengan penggunaan relasi. Contoh: *SELECT* nama_kolom1, nama_kolom2 *FROM* nama_tabel;

c. *UPDATE*
Perintah update digunakan untuk memperbaharui data pada sebuah tabel. Contoh: *UPDATE* nama_tabel *SET* kolom1=data1, kolom2=data2,… *WHERE* kolom=data;

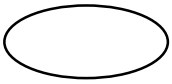
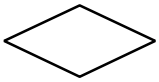
d. *DELETE*
Perintah delete digunakan untuk menghapus data dari sebuah tabel. Contoh: *DELETE FROM* nama_tabel *WHERE* kolom=data;

2.2.9 Entity Relationship Diagram (ERD)

ERD adalah suatu rancangan atau bentuk hubungan suatu kegiatan di dalam sistem yang berkaitan langsung dan mempunyai fungsi di dalam proses tersebut. ERD adalah suatu pemodelan dari basis data relasional yang didasarkan atas persepsi di dalam dunia nyata, dunia ini senantiasa terdiri dari sekumpulan objek yang saling berhubungan antara satu dengan yang lainnya. Suatu objek disebut *entity* dan hubungan yang dimilikinya disebut *relationship*. Suatu *entity* bersifat unik dan memiliki atribut sebagai pembeda dengan *entity* lainnya. Berikut merupakan simbol-simbol dari ERD:

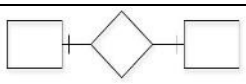
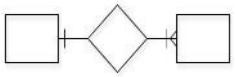
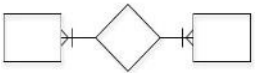
Tabel 2. 3 Simbol ERD

No.	Nama	Simbol	Keterangan Fungsi

1.	Entitas		Persegi panjang menyatakan himpunan entitas adalah orang, kejadian, atau berada.
2.	Atribut		Atribut merupakan informasi yang diambil tentang sebuah entitas.
3.	Relasi		Belah ketupat menyatakan himpunan relasi merupakan hubungan antar entitas.
4.	Link		Garis sebagai penghubung antara himpunan, relasi, dan himpunan entitas dengan atributnya

ERD memiliki derajat relasi atau biasa disebut kardinalitas. Kardinalitas menjelaskan batasan jumlah keterhubungan satu entity dengan entity lainnya.

Tabel 2. 4 Macam-Macam Kardinalitas

No	Nama	Simbol	Keterangan
1	Relasi Satu ke Satu (*One to One*)		Relasi yang menunjukkan bahwa setiap himpunan entitas berhubungan dengan tepat satu himpunan entitas lainnya
2	Relasi Satu ke Banyak (*One to Many*)		Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak, begitu pula sebaliknya
3	Relasi Banyak ke Banyak (*Banyak to Many*)		Relasi yang menunjukkan bahwa setiap himpunan entitas boleh berhubungan dengan banyak himpunan entitas lainnya dan sebaikanya

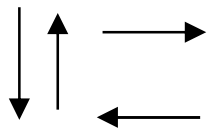
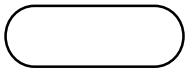
2.2.10 *Flowchart*

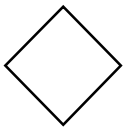
Flowchart adalah bagan (*chart*) yang menunjukan alir (*flow*) di dalam program atau prosedur sistem secara logika. Bagan alir (*flowchart*) digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi. Ada beberapa jenis-jenis *flowchart* diantaranya:

a. Bagan alir sistem (*system flowchart*)
b. Bagan alir dokumen (*document flowchart*)
c. Bagan alir skematik (*schematic flowchart*)
d. Bagan alir program (*program flowchart*)
e. Bagan alir proses (*process flowchart*)

Simbol-simbol dalam *flowchart* dapat dilihat pada Table 2.3

Tabel 2. 5 Simbol Flowchart

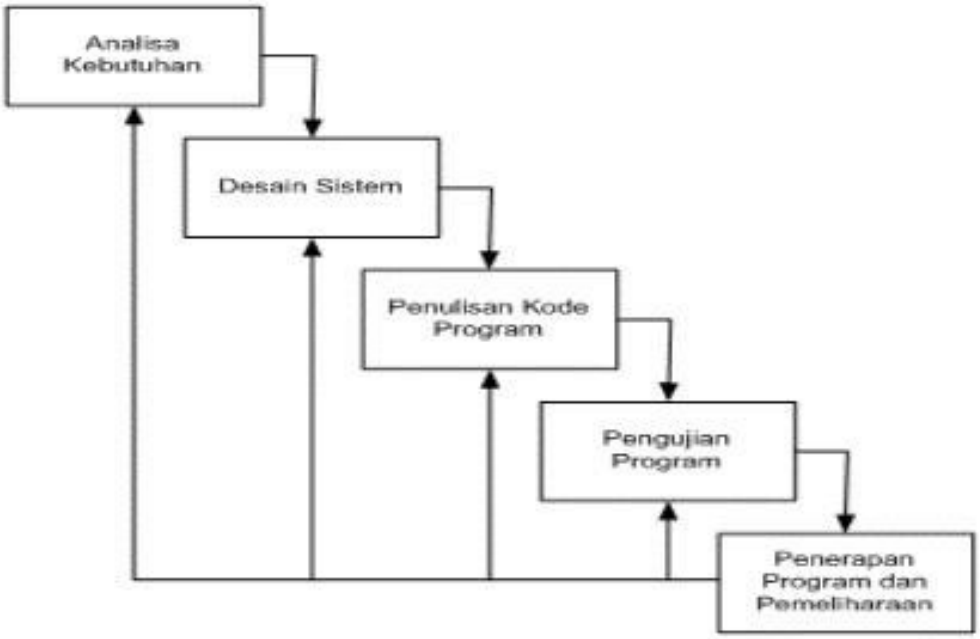
No	Simbol	Keterangan
1	 ***Flow Direction Symbol***	Yaitu simbol yang digunakan untuk menghubungkan antara simbol yang satu dengan simbol yang lain. Simbol ini disebut juga *connecting* line.
2	 ***Terminator Symbol***	Yaitu simbol untuk permulaan (*start*) atau akhir (*end*) dari suatu kegiatan.
3	 ***Processing Symbol***	Yaitu simbol yang menunjukan pengolahan yang dilakukan oleh komputer.

4	 ***Manual Operation Symbol***	Yaitu simbol yang menunjukan pengolahan data yang tidak dilakukan oleh komputer.
5	 ***Decision Symbol***	Yaitu simbol untuk pemilihan proses berdasarkan kondisi yang ada.
5	 ***Input -Output Symbol***	Yaitu simbol yang menyatakan proses *input* dan *output* tanpa tergantung dari jenis peralatannya.
6	 ***Document Symbol***	Yaitu simbol yang menyatakan *input* berasal dari dokumen dalam bentuk kertas atau *output* dicetak ke kertas.

2.2.11 Metode Waterfall

Metode yang digunakan dalam pembuatan sistem ini adalah menggunakan pendekatan berorientasi data atau terstruktur yaitu *Linier Squential Model*, model proses ini sering disebut juga sebagai Metode *Waterfall*.

Menurut Ian Sommerville, *Waterfall* Model atau biasa disebut klasik *Life Cycle* adalah model klasik yang bersifat sistematis, berurutan dalam membangun piranti lunak. Aktivitas-aktivitas dalam *waterfall* model adalah sebagai berikut[12]:



Gambar 2. 1 Metode *Waterfall*

a. Komunikasi *(Communication)*

   Pada tahap ini akan dilakukan inisiasi proyek, seperti menganalisis masalah yang ada dan tujuan yang akan dicapai.

b. Perencanaan *(Planning)*

   Tahap ini merupakan tahap dimana akan dilakukan estimasi mengenai kebutuhan-kebutuhan yang diperlukan untuk membuat sebuah sistem. Selain itu, penjadwalan dalam proses pengerjaan juga ditentukan pada tahap ini.

c. Pemodelan *(Modeling)*

   Kemudian mulai masuk pada tahap pemodelan dimana perancang menerjemahkan kebutuhan sistem kedalam representasi untuk menilai kualitas sebelum tahap selanjutnya dikerjakan.

d. Konstruksi *(Construction)*

   Construction merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemah desain dalam bahasa yang bisa dikenali oleh komputer. Programmer akan menerjemahkan transaksi yang diminta oleh *user*. Tahap ini yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan pengujian (*Testing*) terhadap sistem yang telah dibuat tadi. Tujuan pengujian adalah

menemukan kesalahan-kesalahan terhadap sistem tersebut untuk kemudian bisa diperbaiki [3].

Pengujian sistem dilakukan untuk mengetahui kesalahan yang muncul ketika program selesai dibuat untuk selanjutnya dapat diperbaiki. Pengujian ini secara fungsional dilakukan untuk meminimalkan kesalahan dan memastikan bahwa semua bagian lulus pengujian dan hasil *output* sesuai dengan keinginan pengguna. Pengujian yang dilakukan pada penelitian ini adalah pengujian *black-box* untuk memperlihatkan fungsi perangkat lunak dapat beroperasi dengan baik.

Pengujian *black-box* didasarkan pada detail dari sistem seperti antarmuka, fungsi dan kesesuaian alur dengan permintaan pengguna. Berbeda dengan pengujian *white-box* yang menguji kode program, pengujian *black-box* tidak dapat menguji bahkan melihat kode program. Selain itu, *black-box Testing* juga menguji *input* serta *output* yang dihasilkan dengan menggunakan semua persyaratan fungsional suatu program. Tidak seperti pengujian *white-box*, yang dilakukan pada saat awal proses pengujian, pengujian *black-box* cenderung diaplikasikan pada tahap akhir dikarena pengujian *black-box* memperhatikan struktur - struktur kontrol, sehingga perhatian berfokus pada domain Informasi. Pengujian *black-box* dapat menemukan kesalahan dengan menggunakan kategori sebagai berikut:

1) Fungsi yang tidak benar atau hilang
2) Kesalahan antarmuka
3) Kesalahan struktur data atau akses *database* eksternal
4) Kesalahan kinerja
5) Kesalahan inisialisasi ataupun terminal Pengujian *black-box* lebih baik dilakukan pada tahap akhir.

Penguji harus merancang skenario pengujian yang dapat memberitahu mengenai ada atau tidaknya kesalahan melalui beberapa proses yang ada, diantaranya yaitu:

1) Menganalisa kebutuhan dan spesifikasi dari perangkat lunak
2) Pemilihan jenis *input* yang menghasilakan *output* dengan benar serta *input* yang mungkin menghasilkan *output* yang salah
3) Menentukan *output* untuk setiap *input* -an
4) Pengujian dilakukan dengan *input*-an yang benar-benar telah diseleksi
5) Melakukan pengujian

6) Perbandingan *output* yang dihasilkan dengan *output* yang diharapkan
7) Menentukan fungsional yang seharusnya ada pada perangkat lunak

e. Pengembangan *(Deployment)*

Tahap ini bisa dikatakan final dalam pembuatan sebuah software atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user*. Kemudian software yang telah dibuat dilakukan pemeliharaan secara berkala.

~Halaman ini sengaja dikosongkan~